UNSUPERVISED APPROACHES TO TEXT CORRECTION USING GOOGLE *N*-GRAMS FOR ENGLISH AND ROMANIAN

Diana INKPEN and Aminul ISLAM

University of Ottawa
School of Information Technology and Engineering
E-mail: {diana, mdislam}@site.uottawa.ca

Abstract. We present an unsupervised approach that can be applied to test corrections tasks such as real-word error correction, near-synonym choice, and preposition choice, using *n*-grams from the Google Web 1T dataset. We present in details the method for correcting preposition errors, which has two phases. We categorize the *n*-gram types based on the position of the gap that needs to be replaced with a preposition. We also consider the normalized frequency values of the candidate prepositions. Our experimental results are better than those reported in related work, on the same test set that was also used in related work. We applied our method to English, but it can be easily applied to Romanian. Google released *n*-gram counts for 10 European languages, including Romanian. As test set, a part of a Romanian corpus can be used. A subset of prepositions needs to be chosen. If we include prepositions that consist in two words, the algorithm needs to be adjusted.

Key words: Text correction, Preposition choice, English, Romanian, Google *n*-gram.

1. INTRODUCTION

There are three basic operations for text correction: namely, *replace*, *insert*, and *delete*. For any text correction, one or a combination of these operations needs to be performed. To give an example, consider an input text, "*he wss very good tea teacher*". By preserving the intended or semantic meaning of the input text as much as possible, it is obvious that one of the best candidates for the corrected version would be "*he was a very good teacher*". To have this corrected version of the input text, we need to *replace* '*wss*' by '*was*', to *insert* '*a*' in between '*wss*' and '*very*', and to *delete* the word '*tea*'. The *replace* operation is one of the most used operations for text correction. Spelling error correction, real-word spelling error correction, near-synonym choice, and preposition choice are some of the correction

tasks where solely the *replace* operation is used. To have a clear idea of how the *replace* operation works, we will focus here on one specific correction task, the preposition error correction.

Prepositions are known to be one of the most frequent sources of errors for English. Bitchener *et al*. (2005) found that preposition errors accounted for 29% of all the errors made by intermediate to advanced English as Second Language (ESL) students. As a result, it seems desirable to focus on this problematic and very common part of speech, in developing a system for automatic error correction in English writing.

Prepositions are challenging for learners because they can appear to have an idiosyncratic behavior which does not follow any predictable pattern even across nearly identical contexts. That is, the choice of a preposition for a given context also depends upon the intention of the writer. For example, "they sat near the beach", "at the beach", "on the beach", "by the beach" are all grammatically correct. Prepositions are so difficult to master because they perform so many complex roles. In English, prepositions appear in adjuncts, they mark the arguments of predicates, and they combine with other parts of speech to express new meanings.

The preposition error correction method that we propose here uses the Google Web 1T *n*-gram data Set (Brants & Franz, 2006), contributed by Google Inc., that contains English word *n*-grams (from unigrams to 5-grams) and their observed frequency counts calculated over 1 trillion words from web page text collected by Google in January 2006. The text was tokenized following the Penn Treebank tokenization, except that hyphenated words, dates, email addresses and URLs are kept as single tokens. The sentence boundaries are marked with two special tokens <S> and </S>. Words that occurred fewer than 200 times were replaced with the special token <UNK>. Table 1 shows the data sizes of the Web 1T corpus. The *n*-grams themselves must appear at least 40 times to be included in the Web 1T corpus[1]. It is expected that this data will be useful for statistical language modelling, *e.g*., for machine translation or speech recognition, as well as for other uses.

Although the focus of this paper is on English, our method does not draw any specificity of the language. Our method could be applied, with almost no changes, to many other languages that have enough available *n*-grams. In October, 2009, Google released Web 1T 5-gram, 10 European Languages Version 1 (Brants & Franz, 2009) consisting of word *n*-grams and their observed frequency counts for ten European languages: Czech, Dutch, French, German, Italian, Polish, Portuguese, Romanian, Spanish and Swedish. Thus, it is possible to use the proposed method for these languages.

[1] Details of the Google data set can be found at www.ldc.upenn.edu/Catalog/docs/LDC2006T13/readme.txt

Table 1. Google Web 1T Data Sizes

Number of	Number	Size on disk (in KB)
Tokens	1,024,908,267,229	N/A
Sentences	95,119,665,584	N/A
Unigrams	13,588,391	185,569
Bigrams	314,843,401	5,213,440
Trigrams	977,069,902	19,978,540
4-grams	1,313,818,354	32,040,884
5-grams	1,176,470,663	33,678,504

This paper is organized as follow: Section 2 presents a brief overview of the related work. Our proposed method is described in Section 3. Evaluation and experimental results for English are discussed in Section 4. Section 5 explains how the method can be applied directly to the Romanian language and how to construct a test set for correction preposition errors in Romanian texts. We conclude in Section 6.

2. RELATED WORK

To the best of our knowledge, almost all of the methods for correcting preposition error are based on supervised approaches. A method for correcting preposition errors for French as a second language is presented in (Hermet & Szpakowicz, 2008). It is unsupervised and it uses counts collected from the Web in a simple way, in order to rank the candidates. Eeg-Olofsson & Knutsson (2003) used 31 handcrafted matching rules to detect extraneous, omitted, and incorrect prepositions in Swedish text written by native speakers of English, Arabic, and Japanese. The rules, which were based on the kinds of errors that were found in a training set of text produced by non-native Swedish writers, targeted spelling errors involving prepositions and some particularly problematic Swedish verbs. In a test of the system, 11 of 40 preposition errors were correctly detected.

Izumia *et al*. (2004) train a maximum entropy classifier to recognize various errors using contextual features. Their results for different error types are: for omission – precision 75.7%, recall 45.67%; for replacement – precision 31.17%, recall 8%; but there is no break-down of results by individual parts of speech. Therefore we do not know what the results were for prepositions only. Chodorow *et al.* (2007) present an approach to preposition error detection which also uses a model based on a maximum entropy classifier trained on a set of contextual features, together with a rule-based filter. They report 80% precision and 30% recall. Gamon *et al*. (2008) use a complex system including a decision tree and a language model for preposition errors.

Bergsma *et al*. (2009) present a unified view of web-scale approaches to lexical disambiguation where they use the counts of 5-grams, 4-grams, trigrams and bigrams in a supervised method on the task of preposition selection. They also

use an unsupervised version of their supervised method where they produce a score for each candidate by summing the (unweighted) log-counts of all context patterns (*i.e.*, 5-grams, 4-grams, trigrams and bigrams) using that candidate, and the candidate with the highest score is taken as the solution. While they use the counts of all the *n*-grams ($n \in \{5,4,3,2\}$), we use the counts of any subsequent (*n*-1)-grams only if we do not get any suggestion using the counts of *n*-grams. As a result, our method is computationally more efficient. Their supervised method obtained 75.4% accuracy and unsupervised method obtained 73.7% accuracy. We do not directly compare our results to their results because of the unavailability of their test data set.

Felice & Pulman (2008) propose a classifier-based supervised approach to correct preposition error in native (L1) English and non-native (L2) English that uses a corpus of grammatically correct English to train a maximum entropy classifier on examples of correct usage. The L1 source they use is the British National Corpus (BNC). They represent training and testing items as vectors of values for linguistically-motivated contextual features. Their feature vectors include 13 feature categories for prepositions. We will compare our results to their results, on the same test data.

3. PROPOSED METHOD

Our task is to find the best preposition from a set of candidates that could fill in the gap in an input text, using the Google Web 1T data set. The gap is where a preposition was in the original text. In this case, we know what the expected solution is, the original preposition in this text. First, we use the Google 5-gram data set to find the best choice from a set of candidate prepositions. If the 5-gram data set fails to generate a choice, then we move to the 4-gram data set, the 3-gram data set, or the 2-gram data set, if the preceding data set fails to generate at least one choice. Let us consider an input text W which after tokenization[2] has p ($2 \le p \le 9$) words[3], *i.e.*, $W = \{\cdots w_{i-4} w_{i-3} w_{i-2} w_{i-1} [w_i] w_{i+1} w_{i+2} w_{i+3} w_{i+4} \cdots\}$,

[2] We need to tokenize the input sentence to make the *n*-grams formed using the tokens returned after the tokenization consistent with the Google *n*-grams. The input sentence is tokenized in a manner similar to the tokenization of the Wall Street Journal portion of the Penn Treebank. Notable exceptions include the following:

- Hyphenated words are usually separated, and hyphenated numbers usually form one token.
- Sequences of numbers separated by slashes (*e.g.*, in dates) form one token.
- Sequences that look like urls or email addresses form one token.

[3] If the input text has more than 9 words then we keep at most four words before the gap and at most four words after the gap to make the length of the text 9. We choose these numbers so that we could maximize the number of *n*-grams to use, given that we have up to 5-grams in the Google Web 1T data set.

where $[w_i]$(in position *i*) indicates the gap and w_i in $[w_i]$ denotes a set of *m* prepositions (*i.e.*, $w_i = \{s_1, s_2, \cdots, s_j, \cdots, s_m\}$). We take into account at most four words before the gap and at most four words after the gap. Our task is to choose the $s_j \in w_i$ that best matches with the context. In other words, the position of *i* is the gap that needs to be filled with the best suited member from the set, w_i. First, we discuss how we categorize different *n*-grams based on the gap position and how we find the normalized frequency value. Then, we discuss the procedure to find the best choice preposition using the *n*-gram ($n \in \{5,4,3,2\}$) data set.

3.1. Categorizing *n*-gram type based on the gap position

We categorize an *n*-gram type (we call it, *k*) based on the position of the gap in the *n*-gram. When we consider the 5-gram data set, there might be five types of 5-grams (*i.e.*, $k \in \{1,2,3,4,5\}$) based on the position of the gap in the 5-gram. For example, if the gap position is the last position in the 5-gram then we call it type 1 (*i.e.*, $k = 1$). Thus, a 5-gram, $w_{i-4}w_{i-3}w_{i-2}w_{i-1}\lfloor s_j \rfloor$, could be represented incorporating *k* as $w_{(i-4)k}w_{(i-3)k}w_{(i-2)k}w_{(i-1)k}\lfloor s_{jk} \rfloor$. All the five types of 5-grams are as follows:

$$w_{(i-4)k}w_{(i-3)k}w_{(i-2)k}w_{(i-1)k}\left[s_{jk}\right] \qquad (k=1)$$
$$w_{(i-3)k}w_{(i-2)k}w_{(i-1)k}\left[s_{jk}\right]w_{(i+1)k} \qquad (k=2)$$
$$w_{(i-2)k}w_{(i-1)k}\left[s_{jk}\right]w_{(i+1)k}w_{(i+2)k} \qquad (k=3)$$
$$w_{(i-1)k}\left[s_{jk}\right]w_{(i+1)k}w_{(i+2)k}w_{(i+3)k} \qquad (k=4)$$
$$\left[s_{jk}\right]w_{(i+1)k}w_{(i+2)k}w_{(i+3)k}w_{(i+4)k} \qquad (k=5)$$

Similarly, all the four types of 4-grams are:

$$w_{(i-3)k}w_{(i-2)k}w_{(i-1)k}\left[s_{jk}\right] \qquad (k=1)$$
$$w_{(i-2)k}w_{(i-1)k}\left[s_{jk}\right]w_{(i+1)k} \qquad (k=2)$$
$$w_{(i-1)k}\left[s_{jk}\right]w_{(i+1)k}w_{(i+2)k} \qquad (k=3)$$
$$\left[s_{jk}\right]w_{(i+1)k}w_{(i+2)k}w_{(i+3)k} \qquad (k=4)$$

The three types of 3-grams are as follows:

$$w_{(i-2)k}w_{(i-1)k}\left[s_{jk}\right] \qquad (k=1)$$

$$w_{(i-1)k}\left[s_{jk}\right]w_{(i+1)k} \qquad (k=2)$$

$$\left[s_{jk}\right]w_{(i+1)k}w_{(i+2)k} \qquad (k=3)$$

And the two types of 2-grams are:

$$w_{(i-1)k}\left[s_{jk}\right] \qquad (k=1)$$

$$\left[s_{jk}\right]w_{(i+1)k} \qquad (k=2)$$

3.2. Normalized frequency value

We determine the *normalized frequency value* of each candidate preposition for the gap position with respect to all other candidates. If we have *m* candidate choices for the gap position, *i*, which are $\{s_1, s_2, \cdots, s_j, \cdots, s_m\}$, and their frequencies $\{f_1, f_2, \cdots, f_j, \cdots, f_m\}$, where f_j is the frequency of a *n*-gram (where $n \in \{5,4,3,2\}$ and any candidate preposition s_j is a member of the *n*-gram), then we determine the normalized frequency value of any candidate preposition s_j as the frequency of the *n*-gram containing s_j, over the maximum frequency among all the candidate prepositions for that position.

$$F(s_j) = \frac{f_j}{\max(f_1, f_2, \cdots, f_j, \cdots, f_m)} \tag{1}$$

Now, based on the types of *n*-gram, *k*, equation (1) can be written as:

$$F(s_{jk}) = \frac{f_{jk}}{\max(f_{1k}, f_{2k}, \cdots, f_{jk}, \cdots, f_{mk})} \tag{2}$$

3.3. Determining the best choice preposition (phase 1)

In Phase 1, we first use the Google 5-gram data set to find the best choice preposition. If the 5-gram data set fails to generate a choice then we back off to the 4-gram data set, the 3-gram data set, or the 2-gram data set, if needed. We apply Phase 2 only if the *n*-gram (where $n \in \{5,4,3,2\}$ data set in Phase 1 fails to generate at least one choice.

3.3.1. Determining the best choice preposition using the 5-gram data set

First, we determine f_{11}, which is the frequency of the 5-gram $w_{(i-4)1}w_{(i-3)1}w_{(i-2)1}w_{(i-1)1}[s_{11}]$ with a specific type k = 1, where the last word of the 5-gram is the first preposition choice among the m candidates.

Similarly, we determine f_{j1}, for all $j \in \{2 \cdots m\}$. Now, we determine $F(s_{j1})$ using equation (2), where k = 1. In the same way, we determine $F(s_{j2})$, $F(s_{j3})$, $F(s_{j4})$ and $F(s_{j5})$. Now, the index of the preposition is:

$$j = \begin{cases} j, & \text{if } \left| \arg\max_{j \in \{1 \cdots m\}} \sum_{k=1}^{5} F(s_{jk}) \right| = 1 \\ 0, & \text{if } \left| \arg\max_{j \in \{1 \cdots m\}} \sum_{k=1}^{5} F(s_{jk}) \right| > 1 \end{cases} \quad (3)$$

For simplicity, we assume that $\arg\max_{j \in \{1 \cdots m\}} \sum_{k=1}^{5} F(s_{jk})$ returns the set of values of $j \in \{1 \cdots m\}$ for which $\sum_{k=1}^{5} F(s_{jk})$ for all $j \in \{1 \cdots m\}$ attains its maximum value. If the expression $\sum_{k=1}^{5} F(s_{jk})$ returns 0 for all $j \in \{1 \cdots m\}$, then $\arg\max_{j \in \{1 \cdots m\}} \sum_{k=1}^{5} F(s_{jk})$ will return the set $\{1 \cdots m\}$. Again, if $\sum_{k=1}^{5} F(s_{jk})$ has unique value for more than one candidates, then $\arg\max_{j \in \{1 \cdots m\}} \sum_{k=1}^{5} F(s_{jk})$ will return a set containing the indices of those candidates. Thus, in general, if $\arg\max_{j \in \{1 \cdots m\}} \sum_{k=1}^{5} F(s_{jk})$ returns a single index means that the preposition choice is the word $s_j \in w_i$, we return s_j and exit. If $\arg\max_{j \in \{1 \cdots m\}} \sum_{k=1}^{5} F(s_{jk})$ returns a set of indices containing at least two indices means, then we need to go to the next step to try with 4-grams.

Note that in equation (3), the values of $\sum_{k=1}^{5} F(s_{jk})$ for all $j \in \{1 \cdots m\}$ are known and from that only the j that maximizes $\sum_{k=1}^{5} F(s_{jk})$ is returned. The values of j with their corresponding expression values, $\sum_{k=1}^{5} F(s_{jk})$ for all $j \in \{1 \cdots m\}$, in descending order can be used to score the choices, if required. For our particular task, we use only the j that maximizes $\sum_{k=1}^{5} F(s_{jk})$ to choose the highest scoring preposition.

3.3.2. Determining the best choice preposition using the *n*-gram data set where $n \in \{5,4,3,2\}$

We could generalize equation (3) by the fact that the upper limit of the summation for this equation is actually *n*, where $n \in \{5,4,3,2\}$ is the n-gram that we use. Thus, equation (3) can be generalized as:

$$j = \begin{cases} j, & \text{if } \left| \arg\max_{j \in \{1 \cdots m\}} \sum_{k=1}^{n} F(s_{jk}) \right| = 1 \\ 0, & \text{if } \left| \arg\max_{j \in \{1 \cdots m\}} \sum_{k=1}^{n} F(s_{jk}) \right| > 1 \end{cases} \quad (4)$$

Here, we first try equation (4) with n = 5 and having $j \neq 0$ means that the preposition choice is the word $s_j \in w_i$; we return s_j and exit. Otherwise, we try equation (4) with all possible decreasing n until we get $j \neq 0$, and then we return s_j and exit. Otherwise, we go to phase 2.

3.4. Determining the best choice preposition (phase 2)

The question of why we use phase 2 is best understood by the example "$\cdots$ *and Parchment on Bridgefoot* [ ] *Stratford-upon-Avon, where the barn* $\cdots$ " where the gap, [ ], needs to be filled up by any of {*of*, *to*, *in*, *for*, *on*, *with*, *at*, *by*, *from*}. But, there is no such 5-gram (*e.g.*, *and Parchment on Bridgefoot* [ ], *Parchment on Bridgefoot* [ ] *Stratford-upon-Avon* and so on), 4-gram (*e.g.*, *Parchment on Bridgefoot* [ ], *on Bridgefoot* [ ] *Stratford-upon-Avon* and so on), 3-gram (*e.g.*, *on Bridgefoot* [ ], *Bridgefoot* [ ] *Stratford-upon-Avon* and so on), 2-gram (*e.g.*, *Bridgefoot* [ ]). The reason of the unavailability of such *n*-grams is that "Bridgefoot" is not a very common word in the Google Web 1T data set.

To solve this issue is straightforward. We follow Phase 1 with some small changes: instead of trying to find all the *n*-grams ($n \in \{5,4,3,2\}$) where only $s_{jk} \in w_i$ is changed while keeping all of $\{\cdots, w_{(i-2)k} w_{(i-1)k}, \cdots\}$ unchanged, we try to find all the *n*-grams ($n \in \{5,4,3,2\}$) where $s_{jk} \in w_i$ as well as any but the first member of $\{\cdots, w_{(i-2)k} w_{(i-1)k}, \cdots\}$ are changed while keeping the rest of $\{\cdots, w_{(i-2)k} w_{(i-1)k}, \cdots\}$ unchanged.

4. EVALUATION AND EXPERIMENTAL RESULTS FOR ENGLISH

We restrict our candidate preposition set to the nine most frequent prepositions in the BNC: *of*, *to*, *in*, *for*, *on*, *with*, *at*, *by*, and *from*, same as (Felice & Pulman, 2008) to ensure the conformity, for direct comparison. Felice & Pulman (2008) tested their model on a section of the British National Corpus (BNC) with test set size 536,193. Felice & Pulman (2008) mentioned that their model's performance compared favorably to the best results in the literature, although direct comparisons were hard to draw since different groups trained and tested on different preposition sets and on different types of data (British vs. American English, BNC vs. news reports, and so on). To directly compare with (Felice & Pulman, 2008), we also use the same test set size (536,193 cases) from the BNC. Our best result to date is 75.64% accuracy. Figure 1 relates our results to others reported in the literature on comparable task. The baseline refers to always choosing the most frequent option, namely *of*. It should be noted that (Gamon *et al*., 2008) report more than one figure in their results, as there are two components to their model: one determining whether a preposition is needed, and the other deciding what the preposition should be. The figure reported here refers to the later task, as it is the most similar to the one we are evaluating. Chodorow *et al*. (2007) also discuss some modifications to their model which can increase accuracy; the result noted here is the one more directly comparable to our own approach.

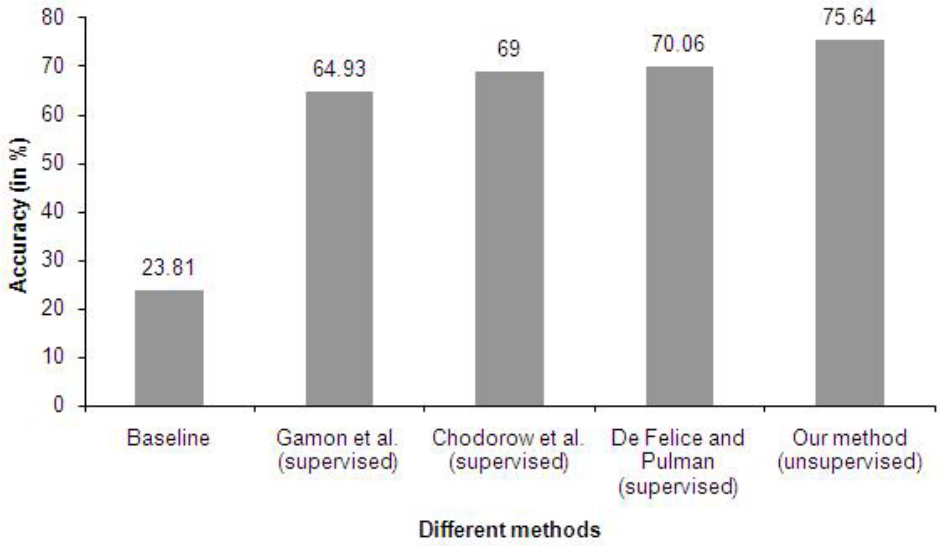

Figure 1. Performance of different methods on L1 prepositions.

4.1. Further discussion and analysis

To assess the method's performance on the L1 data, it is important to consider factors such as performance on individual prepositions, the relationship between test set size and accuracy, and the kinds of errors made by the model.

Table 2 shows the method's performance on individual prepositions together with the size of their test sets. A detailed picture of the method's errors can be observed by looking at the confusion matrix in Table 3, which shows, for each preposition, what the method's incorrect decision was.

Table 2. L1 results – individual prepositions

Prepositions	Test set size	Accuracy
of	135,161	94.45%
to	111,834	86.12%
in	97,558	75.73%
for	42,428	56.81%
with	34,953	57.37%
on	32,628	58.54%
by	31,278	54.44%
at	25,652	63.45%
from	24,701	45.24%

Table 3. Confusion matrix for L1 data – prepositions

Target Prep	Confused with								
	of	to	in	for	with	on	by	at	from
of		17.00%	43.67%	14.67%	7.33%	6.33%	5.00%	3.67%	2.33%
to	35.10%		30.92%	11.92%	7.09%	4.19%	5.15%	2.09%	3.54%
in	51.32%	14.04%		12.25%	5.49%	5.81%	5.60%	2.85%	2.64%
for	32.88%	22.10%	25.78%		7.23%	2.73%	4.09%	2.86%	2.32%
with	32.55%	17.79%	31.71%	8.56%		3.69%	2.52%	1.01%	2.18%
on	30.87%	15.71%	29.02%	5.91%	6.84%		4.25%	3.33%	4.07%
by	33.86%	13.86%	28.60%	8.25%	7.54%	3.16%		2.63%	2.11%
at	18.67%	18.40%	32.00%	11.47%	6.40%	8.00%	2.40%		2.67%
from	29.39%	14.97%	27.73%	8.50%	6.28%	3.51%	5.36%	4.25%	

Table 4 shows some examples of instances where the method's chosen preposition differs from that found in the original text. In most cases, the method's suggestion is also grammatically correct, but the overall meaning of the phrases changes somewhat.

Table 4. Examples of method's errors on L1 task – preposition

Method's choice	Correct phrase
*The connections and friendships **with** Surrealism can also be*	*friendships **of** Surrealism*
*He wants to escape **from** the world*	*escape **to** the world*
*hand were essential ingredients **to** the success of The*	*ingredients **in** the success*
*may not be enough **to** a theoretician.*	*enough **for** a theoretician*
*saw the twentieth century **in** their eyes but they*	*century **with** their eyes*
*figure begins to work **with** us further, now less*	*work **on** us*
*as a sympathetic appraisal **of** a critic who is*	*appraisal **by** a critic*
*Indeed, an article **of** this length will frequently*	*an article **at** this length*
*they seem to proceed **on** his own mind entirely,*	*proceed **from** his own*

For each case, our method returns either a choice (which is either correct or incorrect) or *no suggestion*. Figure 2 shows the number of cases where either a choice (correct or incorrect) or *no suggestion* is generated for different combinations of *n*-grams used. To give an example, using only 5-grams, each of the 316950 cases either generate a correct or an incorrect suggestion. It also means that in the next 5-4-gram combination, we only process the rest of the 219243 cases[4]. Figure 2 validates the intuition behind using a combination of *n*-grams rather than using only *n*-grams (*e.g.*, 5-grams), by showing that while 5-grams generate suggestions for only 316950 cases, a combination of 5-4-3-2-5′-4′-3′-grams[5] generates suggestion for all 536193 cases.

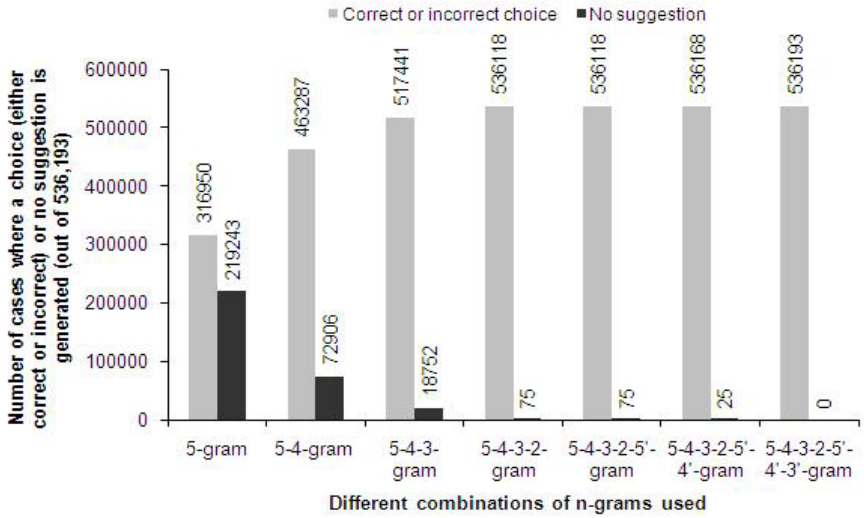

Figure 2. Number of cases where a choice (either correct or incorrect) and *no suggestion* is returned for different combinations of *n*-grams used.

[4] We use the result of the previous 5-grams in 5-4-gram combination, thus we only use 4-grams in this combination.

[5] Apostrophe (′) is used to denote the *n*-grams used in phase 2. *x*-*y*-⋯*z*-gram means that we use *x*-grams, *y*-grams, ⋯ and *z*-grams.

Figure 3 breaks down the numbers shown in Figure 2 into correct choices, incorrect choices and *no suggestion*. To give an example, using only 5-grams, we get correct suggestions for 263638 cases, incorrect suggestions for 53312 cases, and *no suggestion* for 219243 cases; whereas using a combination of 5-4-3-2-5′-4′-3′-grams, we get correct suggestions for 405583 cases, incorrect suggestions for 130610 cases, and *no suggestion* for 0 cases.

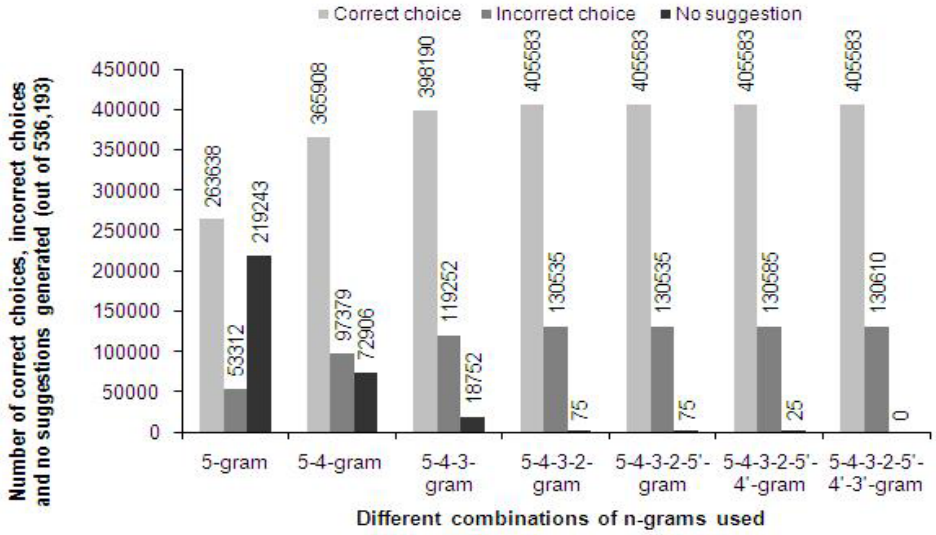

Figure 3. Number of correct choices, incorrect choices and *no suggestions* returned for different combinations of *n*-grams used.

The performance among different combinations of *n*-grams is measured using *Precision* (*P*) and *Recall* (*R*):

$$P = \frac{number\ \ of\ \ correct\ \ suggestions\ \ returned}{number\ \ of\ \ suggestions\ \ returned}$$

$$R = \frac{number\ \ of\ \ correct\ \ suggestions\ \ returned}{total\ \ number\ \ of\ \ cases\ \ in\ \ the\ \ collection}$$

The fraction of suggestions that are correct is the correction precision and the fraction of cases corrected is the correction recall. Figure 4 shows precision and recall for different combinations of *n*-grams used. We get the highest precision (83.18%) when using only 5-grams, which is obvious because 5-grams use the maximum possible context (4 words) and as a result the chance of getting the highest ratio between the number of correct suggestions returned and the number of suggestions returned increases. But the recall at this level is very poor (only

49.17%). Figure 4 demonstrates how recall gets better using different combinations of *n*-grams while keeping precision as high as possible. Using a combination of 5-4-3-2-5′-4′-3′-grams, we achieve equal precision and recall (which is also the accuracy of the method). Thus, the equal precision, recall and accuracy ensure that for each preposition, we get one and only one suggestion. This is not the case for other approaches.

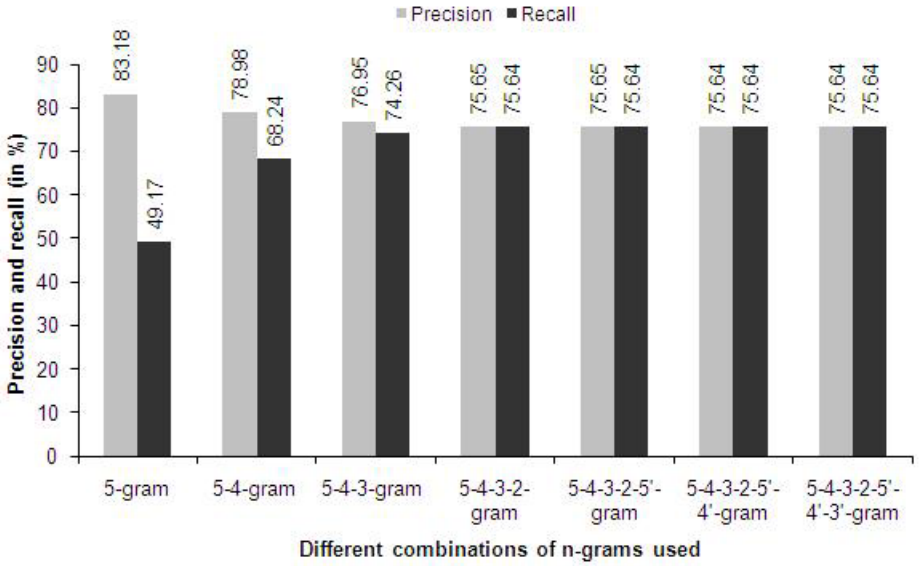

Figure 4. Precision and recall for different combinations of *n*-grams used.

Using a combination of 5-4-3-2-grams, we get a significant improvement of recall, but after that (*i.e.*, a combination of 5-4-3-2-grams to a combination of 5-4-3-2-5′-4′-3′-grams), we do not get any improvement. As a result, the question of whether it makes any sense to try these later combinations arises. We try to answer this issue in Figure 5, which shows the number of cases processed[6] for different combinations of *n*-grams used.

Figure 5 shows that we need to process only 175 more cases when we move from a combination of 5-4-3-2-grams to a combination of 5-4-3-2-5′-4′-3′-grams. Though for this data set we do not get any new correct suggestions, there is always some chance to provide some correct suggestions. Thus, it is worth taking into account these later combinations.

[6] When we use only 5-grams, we process all 536193 cases (where we do not get a suggestion for 219243 cases) and then when we use 4-grams along with 5-grams for 5-4-grams combination, we process these unsolved 219243 cases again, thus totalling the number of cases processed to 755436 for 5-4-grams combination.

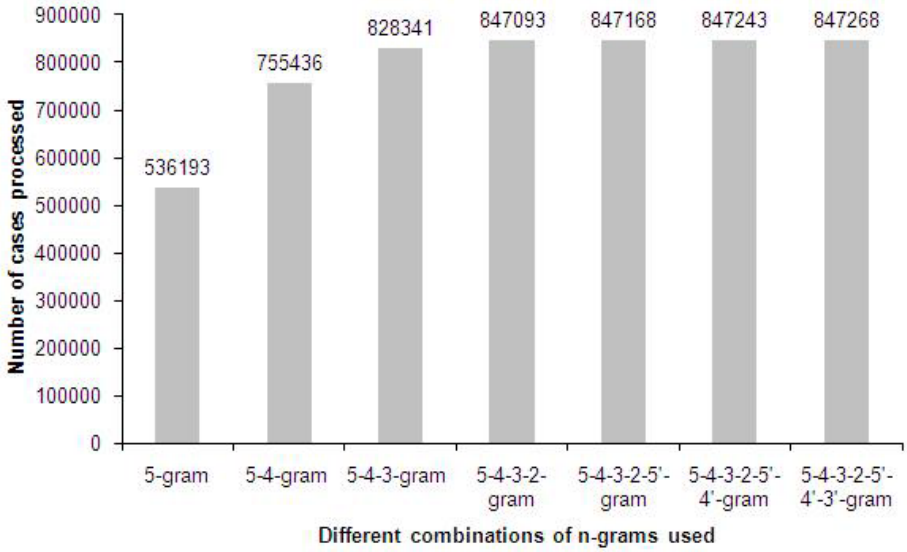

Figure 5. Number of cases processed for different combinations of *n*-grams used.

5. PREPOSITION ERROR CORRECTION FOR ROMANIAN

Google *n*-grams were recently released for 10 European languages[7], including Romanian (Brants & Franz, 2009). The length of the *n*-grams ranges from unigrams (single words) to five-grams. The *n*-gram counts were generated from approximately one hundred billion word tokens of text for each of the 10 languages, or approximately one trillion total tokens. The *n*-grams were extracted from publicly-accessible web pages from October 2008 to December 2008. This data set contains only *n*-grams that appeared at least 40 times in the processed sentences. Less frequent *n*-grams were discarded.

The Romanian *n*-grams from this data set can be used in order to determine the best preposition choices for Romanian texts, in the same way we used the English *n*-grams in the previous sections.

A test set for preposition error correction for Romanian can be collected from existing Romanian corpora. It could be, for example, a part of the corpus from Rada Mihalcea list of resources for Romanian language[8] (this corpus has 50 million words in total) or Orwell's "1984" novel tagged with part-of-speech information (MULTEXT-EAST project[9]).

[7] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2009T25
[8] http://www.cse.unt.edu/~rada/downloads.html
[9] http://nl.ijs.si/ME/

The preposition set needs to be chosen. Here are some examples of prepositions for Romanian:

de/ pe/ in/ cu/ la/ pentru/ dupa/ din/ langa/ spre/ catre/ impotriva/ contra/ ...
de langa/ de dupa/ de la/ de pe/ ...

(for a complete list of all the prepositions in Romanian language see http://www.scritube.com/literatura-romana/gramatica/Prepozitii-si-grupuri-prepozit1351620172.php.)

For initial experiments, we recommend restricting to the top-most frequent prepositions, composed of only one word. This would be similar to the above experiments for English, which focused on 9 prepositions. Our method can be applied with no changes for one-word prepositions. If we include prepositions that consist of two words, the algorithm needs to be adjusted.

6. CONCLUSION

We presented an unsupervised statistical method of correcting preposition errors. We compared this method with three previous supervised methods and show that the performance is comparable or even better. For proprietary reason, we cannot test our method to the L2 data set that Felice & Pulman (2008) and Chodorow *et al*. (2007) use. It has been estimated that about 750M people use English as a second language, as opposed to 375M native English speakers (Crystal, 1997), while as much as 74% of writing in English is done by non- native speakers (Gamon *et al*., 2008). Because the Google Web 1T data set is a representation of both native and non-native English, we can say that our proposed unsupervised method is also equally applicable to L2 English texts. In future, we plan to test our method on a L2 data set.

REFERENCES

1. BERGSMA S., LIN D., and GOEBEL R., Web-scale n-gram models for lexical disambiguation, *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI-09)*. (Pasadena, California), pp. 1507–1512, July 2009.
2. BITCHENER J., YOUNG S., and CAMERON D., The effect of different types of corrective feedback on ESL student writing. *Journal of Second Language Writing*, vol. 14, pp. 191–205, 2005.
3. BRANTS T. and FRANZ A., *Web 1T 5-gram corpus version 1.1*, Technical report, Google Research, 2006.
4. BRANTS T. and FRANZ A., *Web 1T 5-gram, 10 European languages version 1*, Technical report, Linguistic Data Consortium, Philadelphia, 2009.
5. CHODOROW M., TETREAULT J. R., and HAN N.-R., Detection of grammatical errors involving prepositions, in *SigSem '07: Proceedings of the Fourth ACL-SIGSEM Workshop on Prepositions*, (Morristown, NJ, USA), pp. 25–30, Association for Computational Linguistics, 2007.
6. CRYSTAL D., *English as a Global Language*, Cambridge University Press, 1997.

7. EEG-OLOFSSON J. and KNUTSSON O., Automatic grammar checking for second language learners – the use of prepositions. *Proceedings of the 14th Nordic Conference on Computational Linguistics (NoDaLiDa 2003)*, (Reykjavik, Iceland), 2003.
8. FELICE R. D. and PULMAN S. G., A classifier-based approach to preposition and determiner error correction in L2 English. *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, (Manchester, UK), pp. 169-176, August 2008.
9. GAMON M., GAO J., BROCKETT C., KLEMENTIEV A., DOLAN W. B., BELENKO D., and VANDERWENDE L., Using contextual speller techniques and language modeling for ESL error correction. *Proceedings of the Third International Joint Conference on Natural Language Processing (IJCNLP 2008)*, pp. 449-456, 2008.
10. HERMET A. D. M. and SZPAKOWICZ S., Using the web as a linguistic resource to automatically correct lexico-syntactic errors. *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, (Marrakech, Morocco), May 2008.
11. IZUMIA E., UCHIMOTOA K., and ISAHARAA H., SST speech corpus of Japanese learners English and automatic detection of learners errors. *ICAME Journal*, vol. 28, pp. 31-48, 2004.